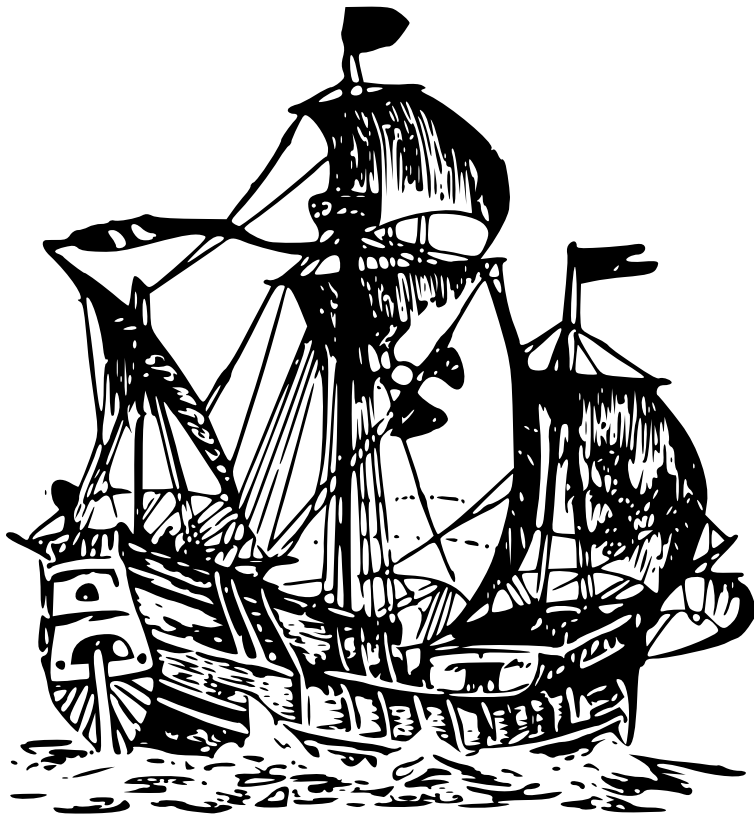




a gentle introduction into a microframework
with good intentions

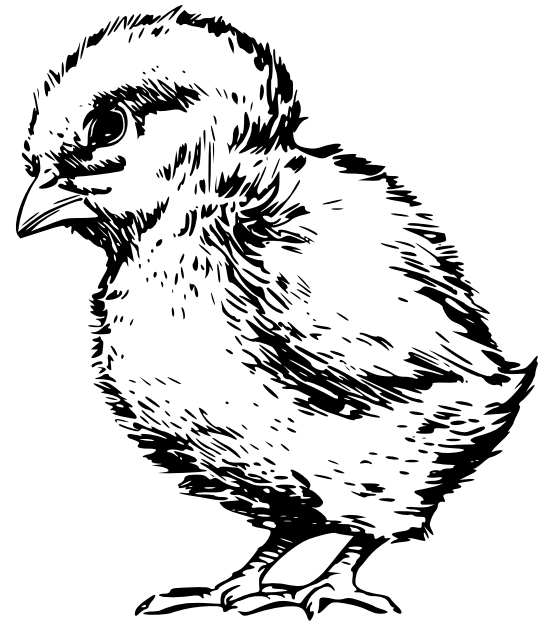
What is Flask?



- » A microframework
- » Reusing existing code
- » Lots of documentation
- » Neat way to write small apps

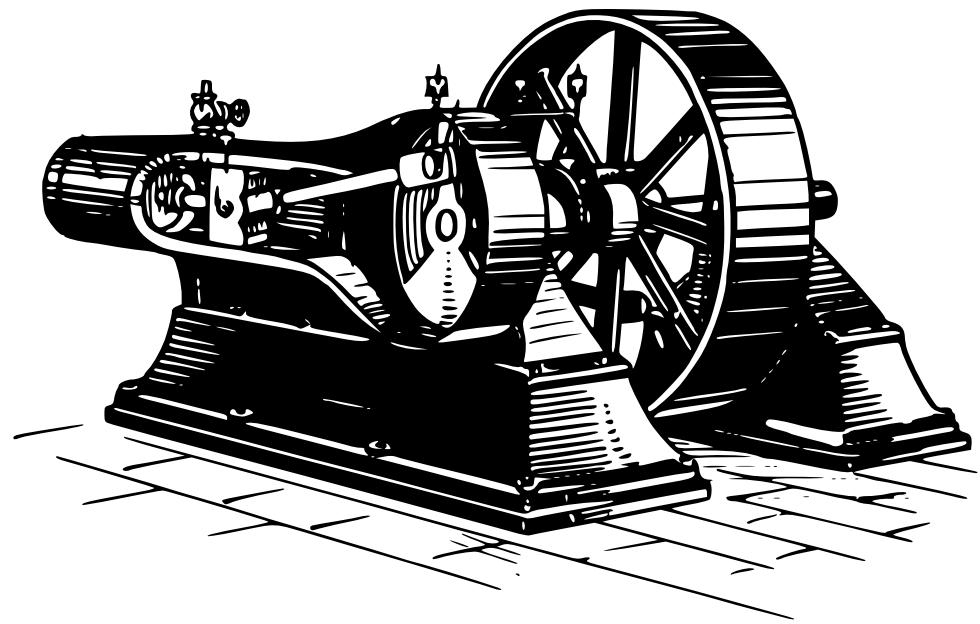
Another μ Framework?

YES!



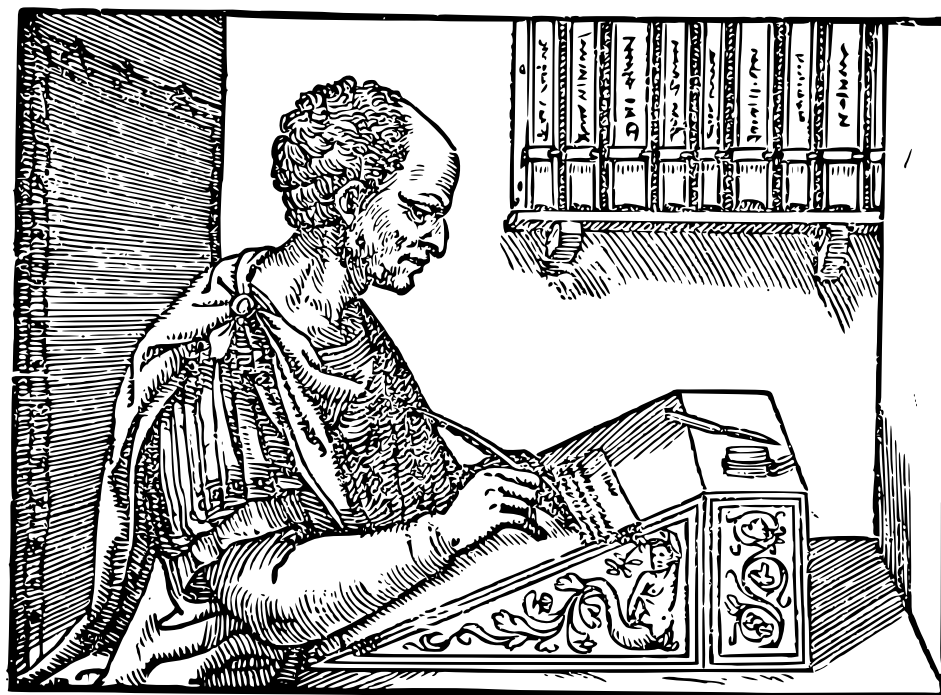
Under the Hood

- » Full power of Werkzeug
- » Jinja2 as a capable template engine



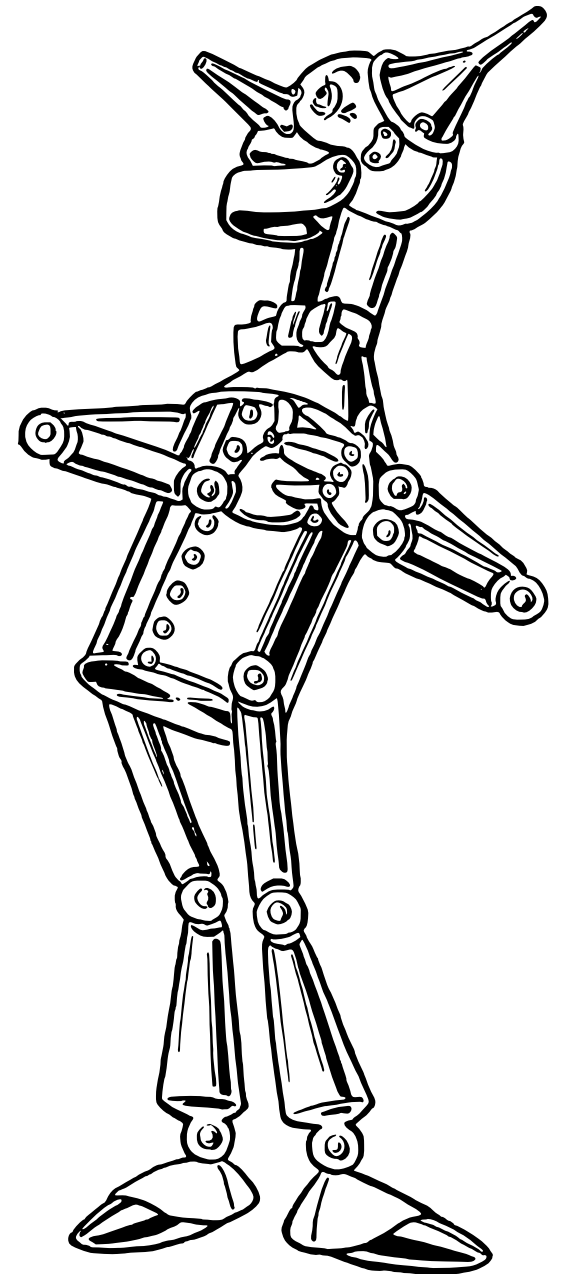
Under the Hood

- » 450 Lines of actual Code
- » 1000 Lines of Tests
- » 5000 Lines of Documentation



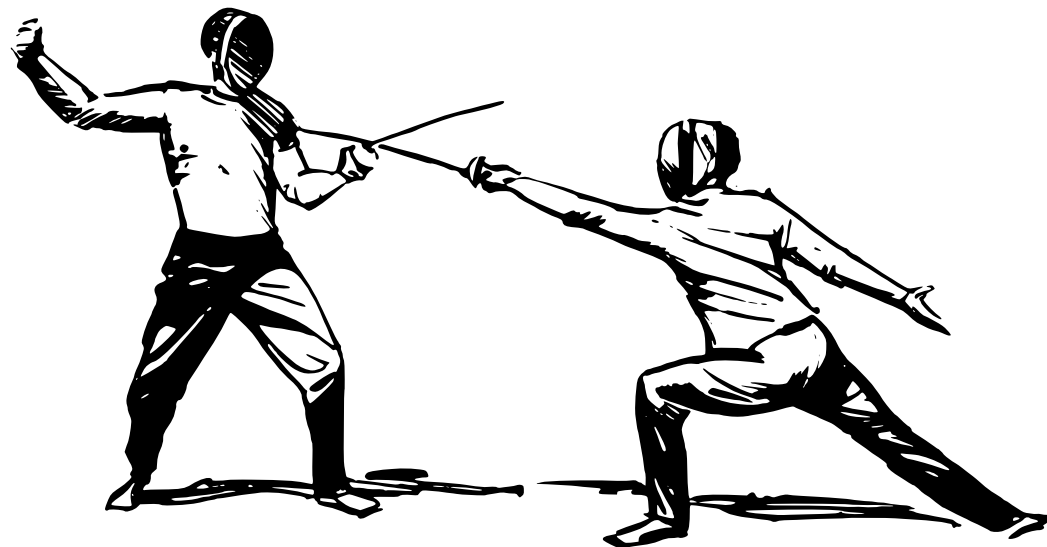
What does it do?

- » Cookie-based session support
- » Flashing of messages
- » Preconfigured Jinja2 with autoescaping
- » Serves static files from “static”
- » Before/After Request hooks
- » Context local objects
- » *RESTful* URL mapping



What else?

- » Lots of documentation (120 A4 pages)
- » Website with lots of snippets
- » Extension registry (OAuth, OpenID, XML-RPC, CSRF protection ...)
- » Active Mailinglist and IRC Channel

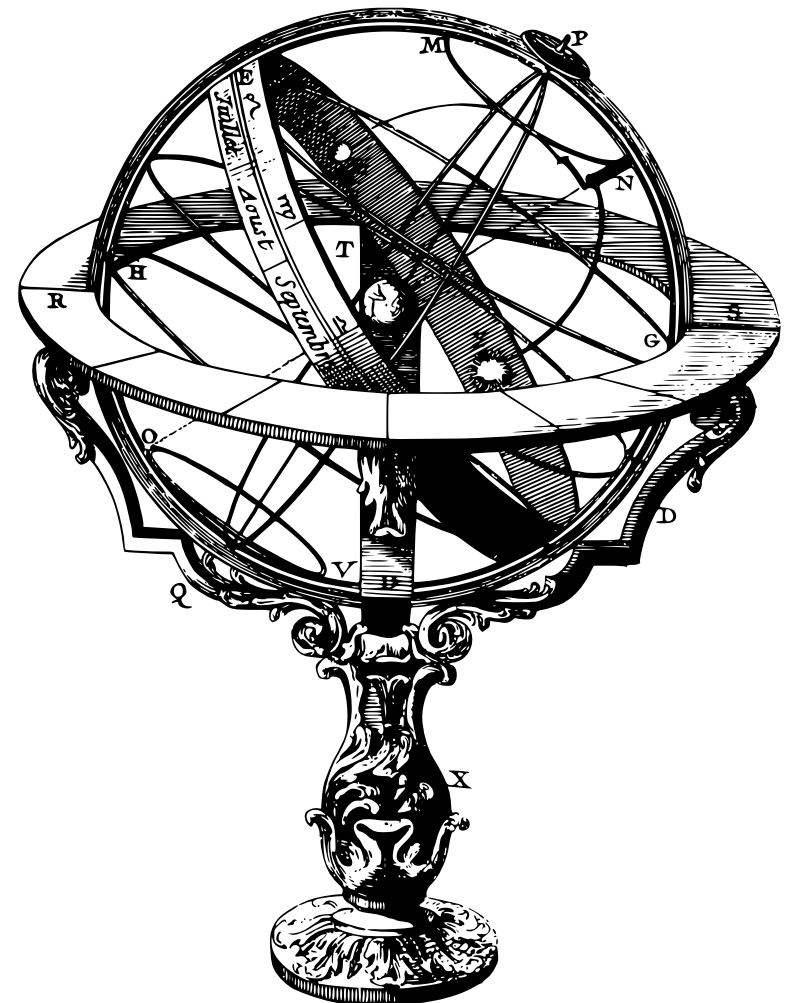


Hello Flask

```
from flask import Flask  
app = Flask(__name__)
```

```
@app.route('/')  
def index():  
    return 'Hello World!'
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```



Hello Localhost

```
$ python hello.py  
* Running on http://127.0.0.1:5000/  
* Restarting with reloader...
```



Rendering Templates

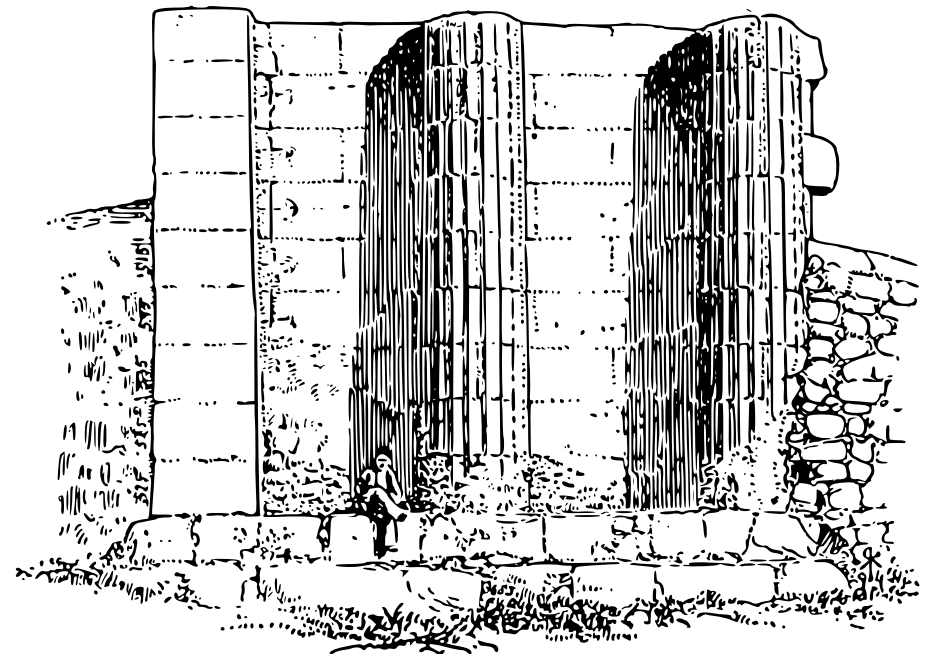
```
from flask import render_template
```

```
@app.route('/')  
def index():
```

```
    return render_template('index.html',
```

```
        variable='value'
```

```
)
```



The Request Data

```
from flask import request, flash, redirect, \
    url_for, request

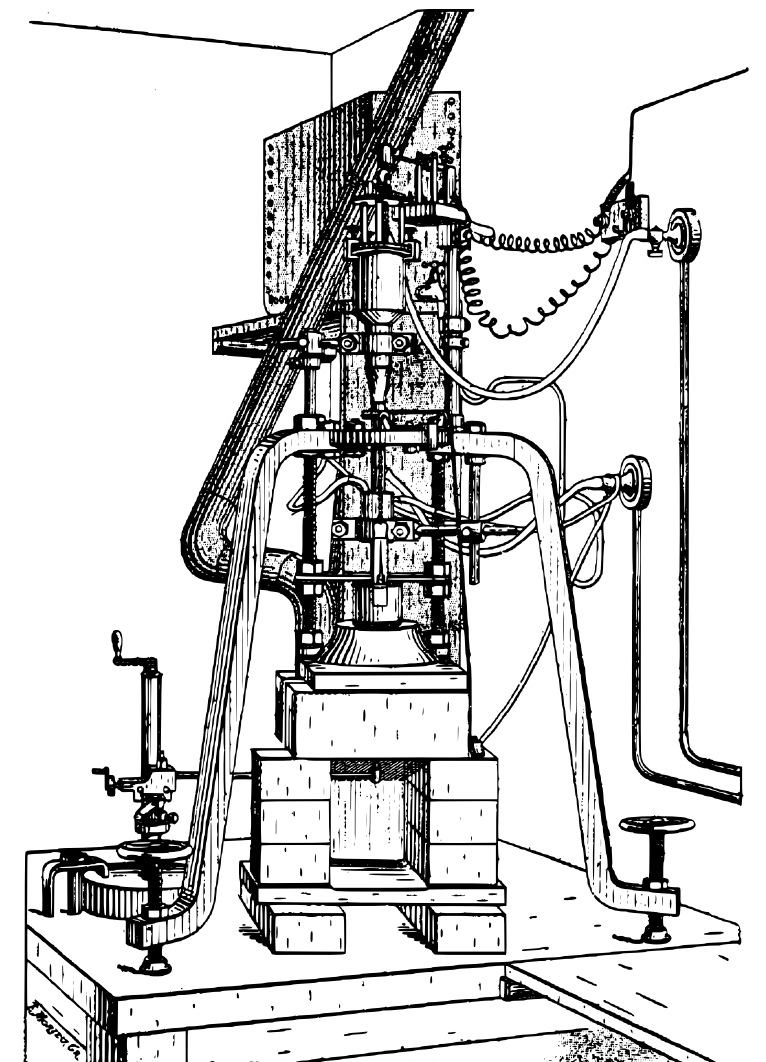
@app.route('/new-comment', methods=['GET', 'POST'])
def new_comment():
    if request.method == 'POST':
        Comment(request.form['name'],
                request.form['text']).save()
        flash('Comment was added')
        return redirect(url_for('show_comments'))
    return render_template('new_comment.html')
```

Before/After Request

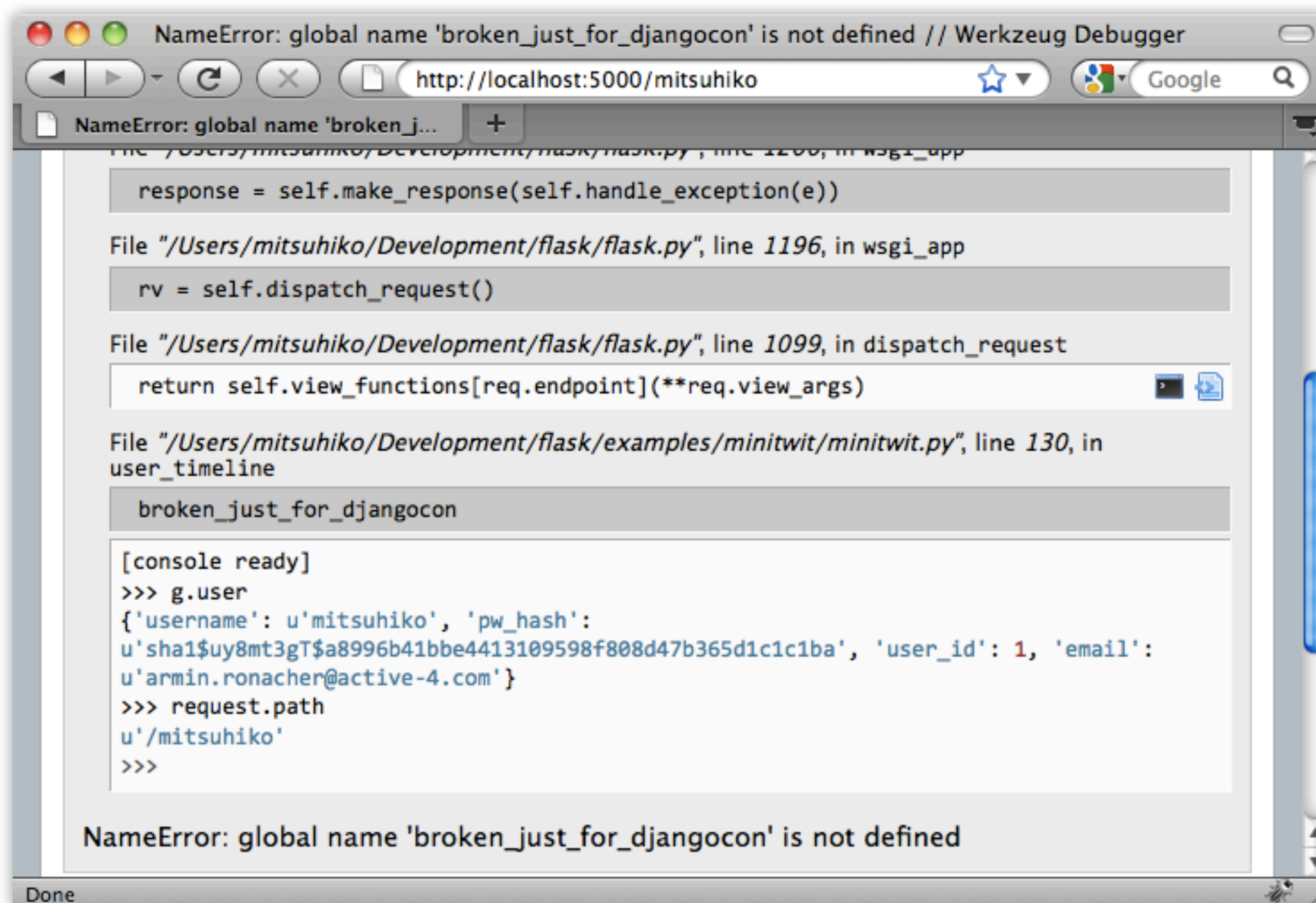
```
import sqlite3
from flask import g

@app.before_request
def before_request():
    g.db = sqlite3.connect(...)

@app.after_request
def after_request(response):
    g.db.close()
    return response
```



If Things Break



The screenshot shows a web browser window with the URL `http://localhost:5000/mitsuhiko`. The browser's developer tools are open, displaying the Werkzeug Debugger. The error message at the top reads: `NameError: global name 'broken_just_for_djangocon' is not defined // Werkzeug Debugger`. The stack trace shows the error occurred in `File "/Users/mitsuhiko/Development/flask/examples/minitwit/minitwit.py", line 130, in user_timeline`. The code snippet for `user_timeline` is shown, with `broken_just_for_djangocon` highlighted. Below the stack trace, the debugger's interactive console shows the following session:

```
[console ready]
>>> g.user
{'username': u'mitsuhiko', 'pw_hash':
u'sha1$uy8mt3gT$a8996b41bbe4413109598f808d47b365d1c1c1ba', 'user_id': 1, 'email':
u'armin.ronacher@active-4.com'}
>>> request.path
u'/mitsuhiko'
>>>
```

At the bottom of the debugger window, the error message is repeated: `NameError: global name 'broken_just_for_djangocon' is not defined`. The status bar at the bottom left of the browser window says "Done".

Where to get?

```
$ pip install Flask
```



<http://github.com/mitsuhiko/flask>

<http://flask.pocoo.org/>